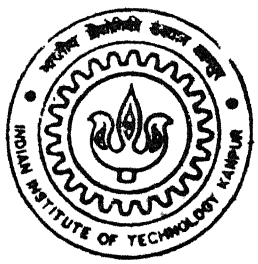
ISOLATED WORD RECOGNITION SYSTEM BASED ON DISCRETE HMM

by
ANIL KUMAR AHIRWAR



DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR

January, 2000

ISOLATED WORD RECOGNITION SYSTEM BASED ON DISCRETE HMM

A Thesis Submitted

in Partial Fulfillment of the Requirements

for the Degree of

MASTER OF TECHNOLOGY

by

ANIL KUMAR AHIRWAR

to the

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

Jan 2000

CERTIFICATE

It is certified that the work contained in the thesis entitled **Isolated word recognition system based on discrete HMM, by Anil Kumar Ahirwar,** has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Dr. G. C. Ray

Professor

Department of Electrical Engineering

Indian Institute of Technology, Kanpur

Jan 2000

Acknowledgement

The author, with immense pleasure expresses his indebtedness and a deep sense of gratitude to Dr. G. C. Ray, whose devotion and encouragement at every step with active participation enabled him to carry out this modest piece of work. It has been a great pleasure for the author to work under him.

Particular thanks are due to Mr. Rohit Sinha for his timely and valuable suggestions. The author wishes to extend his gratitude to Mr. Alot Katiyar, Mr.Ramesh and Mr D.K. Rai for their help. Finally the author wishes to thank his friends in particular Pankaj, Naresh, Ajay, Pawan and Rishi for their cooperation and assistance throughout his stay at IITK.

Anil Kumar Ahirwar

IIT Kanpur

Jan 2000

ABSTRACT

The solution of speaker independent isolated word recognition using vector quantization and hidden *Markov* model based analysis along with front end processing is presented in this thesis. Both the vector quantizer and the hidden *Markov* models need to be trained for the vocabulary to be recognized. In this case such training has resulted in a distinct hidden Markov model for each word in a vocabulary . Recognition consist of computation of probability for each word and selecting the highest.

In this thesis linear predictive coding (LPC) analysis is done in the front end processor to convert the speech signal of a frame into some parametric representation (cepstral coefficient). This results in a series of vectors characteristic of time varying spectral parameters of the speech signal. These vectors are grouped into discrete sets by k-means clustering algorithm. During the training process, different HMMs are modeled for different words in vocabulary. During the recognition process Viterbi algorithm is used to determine the HMM within the sets of HMMs that best matches with the observation sequence.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Real world processes generally produce observable output which can be characterized as signal. The signals may be discrete in nature (e.g character from a finite alphabet, quantized vector from a code book, etc) or continuous (e.g speech samples, temperature measurement etc). The signal source may be stationary (i.e. its statistical properties do not vary with time) or non stationary. The signal may be pure (i.e. coming out from a signal source) or corrupted from other signal sources (e.g noise) or by transmission distortions. The speech signal production may be viewed as a form of filtering in which a sound source excites a vocal tract filter.

The speech signal is slowly time varying signal in the sense that, when examined over a sufficiently short period of time (between 5 and 100msec), its characteristics are fairly stationary. However, over long period of time (on an order of 1/5 seconds or more) the signal characteristics change. Speech signal is considered non stationary in the sense that its characteristics vary with time[1]. Certain characteristics of speech signal are worth mentioning:

- Voiced sounds during vowels are characterized by quasi-periodicity, low frequency content and large amplitude.

- Unvoiced sounds and fricatives are characterized by randomness, high frequency content, and relatively low amplitude.

- The transition between voiced and unvoiced sound is gradual.

To classify the types of signal model, we have to first characterize the time varying signal into distinct classes. There are broadly two different kind of signal classes, deterministic signal and random signal. In a deterministic signal there is no uncertainty with respect to its value at any time i.e these signals may be modeled as completely specified function of time. On the other hand in a random signal, there is some degree of uncertainty before it actually occurs. Based on this classification, a signal model may be characterized as deterministic model and statistical model. In the case of deterministic model, the specification of signal model requires characteristic properties such as amplitude, frequency, phase of sine wave; amplitude and rate of exponentials. In the case of statistical model, one tries to model the statistical nature of the signal. In this case, the signal is assumed to be non-stationary and possesses some well defined parameters which can be estimated in well defined manner.

Signal processing is the first step used to solve the speech recognition problem. In this process relevant information from the speech signal is extracted in an efficient and robust manner. During this step, spectral analysis is used to characterize the time varying properties of the speech signal. The properties of speech signal are stationary for a very small duration of time and they change over a long interval. The short time properties of signal can be conveniently represented by the spectral measurement vectors. There are many ways by which the spectral analysis can be performed. This includes standard methods as measurement of the discrete FFT, LPC, autoregressive/moving average (ARMA)[9] etc. In speech modeling we often call this short time spectral

vector an observation. The short time spectra of a signal can be described in a mathematically consistent framework, thereby offering analytic solution to speech problem.

There are various types of approaches by which isolated word recognition can be done. These approaches can be broadly classified as follows: the acoustic phonetic approach, the pattern recognition approach and neural network based approach[4]. The acoustic phonetic approach is based upon the assumption that there exist finite distinctive phonetic units in the spoken language. It requires extensive knowledge of acoustic properties of phonetic units. Implementation of isolated word recognition system based on this approach is considered in detail by Rabiner in [4].

In pattern recognition based approach we don't bother about the acoustic properties of the phonetic units. In this approach, the model parameters are computed during the training phase. How the actual training is done, is explained in chapter 3. During the recognition phase classification of test word is done by pattern comparison. In conventional pattern recognition system the unknown test token (i.e. pattern) is time aligned in turn to each reference pattern via some form of time wrapping procedure, typically, dynamic time wrapping (DTW)[11, 6]. Complete function of such conventional recognition system is explained as follows. The input speech signal after passing through the band pass filter is digitized. Then the digitized signal is processed through the pre-processing block which provides high frequency pre-emphasis to the speech. Then the pre-emphasized signal is blocked into frames and LPC analysis is performed on each frame of word thus creating test pattern. This test pattern is compared with each reference pattern (obtained via training algorithm) using DTW[1] (dynamic time wrapping) alignment algorithm that

[1]see appendix A.

simultaneously provides a distance score associated with the alignment. The distance scores for all the reference pattern are sent to a decision rule, which provides the classification of spoken word. The pattern recognition approach to speech recognition has following features:

1. The method is easy to understand and rich in mathematics.

2. The performance of the system is sensitive to the amount of training data available.

3. No speech-specific knowledge is used explicitly in the system.

4. It provides high performance.

Itakura [17] introduced the DTW for non linear alignment of speech. This system was tested on a single speaker and achieved 97.3% accuracy on 200-word isolated word task without use of grammar.

Another statistical approach to word recognition uses the HMM. No such direct alignment is performed in HMM system, only an indirect time alignment is obtained based on probability scoring. Rabiner, Levision and Sondhi compared the performance of LPC/DTW and HMM/VQ[6]. A test set of data consisting of one replication of each of the ten digits by a set of 100 talker was used. They found that average word accuracy with LPC/DTW isolated word recognizer was 98.5% and with HMM/VQ it was 96.3%.

Nowadays an approach based on neural network is also popular. A neural network, which is basically parallel distribution processing model, is a dense interconnection of simple, non-linear, computational elements [4]. Four model characteristics must be specified to implement an arbitrary neural network.

1. number and type of input-The issues involved in the choice of inputs to a

neural network are similar to those involved in the choice of features for any pattern classification system.

2. connectivity of the network-This issue involves the size of the network. i.e the number of hidden layers and the number of nodes in each layer between input and output and the type of interconnection.

3. choice of the offset-The choice of the threshold, for each computational element must be made as the part of the training procedure, which chooses values for the interconnected weights and the offset.

4. choice of nonlinearity-exact choice of nonlinearity is also very important in term of network performance. However, non linearity must be continuous and differentiable for the training algorithm to be applicable.

Neural network possess many features which make it attractive for speech recognition. Some of the features are given below:

1. They can readily implement a massive degree of parallel computation.

2. They intrinsically possess a great deal of robustness or fault tolerance.

3. The connection weights of the network need not be constrained to be fixed: they can be adopted in real time to improve performance.

4. Because of the non-linearity within each computational element, a sufficiently large neural network can approximate any non-linearity or non-linear dynamical system.

To be useful for speech recognition, a layered feed forward neural network must have a number of properties. First, it should have multiple layers and sufficient interconnections between units in each of these layers.This is to ensure

that the network will have the ability to learn complex nonlinear decision surface. Second, the network should have the ability to represent relationships between events in time. Third, the actual feature or abstraction learned by the network should be invariant under translation in time. Fourth, the learning procedure should not require precise temporal alignment of the labels that are to be learned.

Time Delay Neural Network (TDNN) architecture satisfies all these criteria. It is simplest Neural Network structure that incorporates speech pattern dynamics. More about TDNN is discussed by Waibel, Hanazawa, Hinton and Lang in [5]. They compared the performance of TDNN with HMM for the 1946 testing token obtained from three speakers and found that TDNN achieves a recognition rate of 98.5% correct while the rate achieved by the HMM was 93.7%.

We deal only with the implementation of isolated word recognition using HMM. Organization of this thesis work is as follows. In chapter 2 basics of hidden Markov model are enumerated, including its three problems. In chapter 3 strategy of processing of signal is explained and how the viterbi algorithm be applied for scoring is shown. In the last chapter results and the model parameters are discussed. In appendix A features of DTW algorithm are explained and, in appendix B theory of LPC is given.

Chapter 2

Basics of hidden Markov model

Hidden Markov model is a widely used statistical method for characterizing the spectral properties of the frame of a signal pattern. These models are also referred to as Markov source or probabilistic function of Markov chain in communication literature. The states in a hidden Markov model are associated with a set of discrete symbols, with an observation probability assigned to each symbol, or are associated with a set of continuous observation with a continuous observation density function. Each transition in a state diagram of an hidden Markov model also has transition probability associated with it.

The foundation of HMM methodology is built on the well known established field of statistics and probability theory. Basic theoretical strength of HMM is that it combines modeling of stationary stochastic processes (for the short time spectra) and the temporal relationship among the processes (via a Markov Chain) together in a well defined probability space. All the features and their underlying basic concepts are discussed in this chapter, starting with the concept of Markov Process.

2.1 Discrete-Time Markov Processes

Discrete time Markov process is the basic concept of hidden Markov model. Discrete time Markov process for N (number of states) equal to five states is explained as follows. Consider a system that may be described at any time as being in one of a set of N distinct states indexed by i {i = 1,2,. . . .N}, as shown in fig 2.1. At regularly spaced, discrete times, the system undergoes a change of state according to a set of probabilities associated with the state. Let q_t denotes a state at the time t. In the case of Markoven process transition from one state to another depends upon just previous state only i.e transition from state q_{t-1} to state q_t is independent of previous states q_{t-2}, q_{t-3} and so on. Mathematically it can be written as

$$P[q_t = j|q_{t-1} = i, q_{t-2} = k, ...] = P[q_{t=}j|q_{t-1} = i] \tag{2.1}$$

Let the transition probability from q_{t-1} to q_t is represented by a_{ij} .i.e state transition probability is the probability that the system is in state j at time instant t, given that the system is in state i at time t-1.

$$a_{ij} = P[q_t = j|q_{t-1} = i] \quad 1 \leq i, j \leq N \tag{2.2}$$

State transition probability has following properties

$$a_{ij} \geq 0 \qquad \forall j, i \tag{2.3}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \quad \forall i \tag{2.4}$$

since they obey standard stochastic constrains.

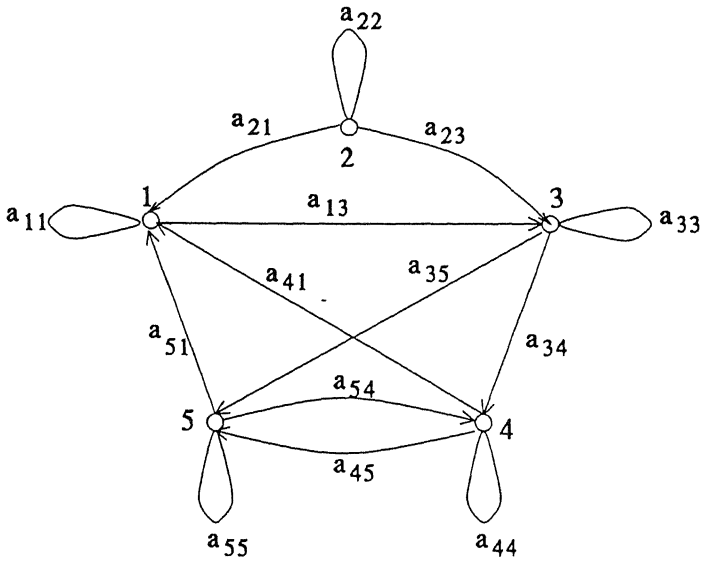


Figure 2.1: Markov chain

The above five states Markov chain can be represented by the matrix given below.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix} \tag{2.5}$$

2.2 Type of HMM

In an ergodic or fully connected HMM every state of the model could be reached (in a single step) from every other state of the model. Transition matrix for such a model having number of states equal to three is shown below

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{2.6}$$

For some application other type of HMMs have been found to perform better than the standard ergodic model. One such model is a left right model. This model has the property that as the time increases, the state index increases i.e state is proceed from left to right. The fundamental property of all left right HMMs is that state transition coefficients have the property

$$a_{ij} = 0 \qquad j < i \tag{2.7}$$

i.e no transitions are allowed to states whose indices are lower than the current state. Further more, the initial state probabilities have the property

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases} \tag{2.8}$$

The state transition matrix for such left right model is thus

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix} \tag{2.9}$$

For the last state in a left right model, the state transition coefficients are specified as

$$a_{NN} = 1, \tag{2.10}$$

$$a_{Nj} = 0 \qquad j \leq N \tag{2.11}$$

Here we have dichotomized HMMs into ergodic and left right models, there are many possible variations and combinations possible. These variations are obtained by placing some constrains on transition probability matrix.

2.3 Elements of HMM

Hidden Markov model is characterized by its three model parameters i.e state transition probability, observation symbol probability and initial state probability. We formally define the elements of HMM as:

- N, the number of states in the model. In HMMs we shall not rigorously define what a state is, but simply say that within the state the signal possesses some measurable and distinctive properties. The individual state are labeled as {1,2,3. . . N} and the state at any time t is denoted as q_t.

- At each clock time t, a new state is entered based upon a transition probability distribution which depends on just previous state (the Markoven property). The state transition probability distribution is represented as A=$\{a_{ij}\}$where

$$a_{ij} = P[q_{t+1} = j | q_t = i], \quad 1 \leq i,\ j \leq N \tag{2.12}$$

- M, denotes the number of distinct observation symbols per state -i.e the discrete alphabets size. The discrete symbol is represented as V=$\{v_1, v_2, \ldots v_M\}$. The observation sequence produced by a system is represented by $O = \{o_1, o_2, \ldots o_T\}$where o_t corresponds to one of the symbol from the discrete symbol set V.

- After each transition is made, an observation output symbol o_t is produced according to observation probability distribution B=$\{b_j(k)\}$. This probability distribution is held fixed for the state regardless of when and how the state is entered. The probability of emitting a symbol v_k at time

t , in the state j is denoted as

$$b_j(k) = P[o_t = v_k | q_t = j], \quad 1 \leq k \leq M \tag{2.13}$$

- The initial state distribution $\pi = \{\pi_i\}$ in which

$$\pi_i = P[q_1 = i], \quad 1 \leq i \leq N. \tag{2.14}$$

Hence the complete specification of an HMM requires: specification of two parameters, N and M , specification of observation symbols, and the specification of the three sets of probability measures A,B, and π . The complete set of parameter is represented as

$$\lambda = (A, B, \pi) \tag{2.15}$$

for a model.

2.4 Three basics problems for HMM

There are three basic problems of interest that must be solved for the model to be useful in real -world applications[2]. These problems are:

- Problem 1: Given the observation sequence $O = o_1 o_2 \ldots o_T$, and a model $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O|\lambda)$, i.e the probability of the observation sequence, given the modal ?

- Problem 2 : Given the observation sequence $O = o_1, o_2, \ldots o_T$ and the model λ, how do we choose a corresponding state sequence $q = q_1 q_2 \ldots q_T$

which is optimum in some meaningful sense (i.e best "explains" the observation)?

- Problem 3 : How do we adjust the model parameter $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$?

Problem 1 is the evaluation problem. The problem is one of scoring, how well a given model matches a given observation sequence. Problem 2 is the one in which we attempt to uncover the hidden part of the model i.e., to find the correct state sequence. Problem 3 is one in which we attempt to optimize the model parameters to best describe how a given observation sequence comes about.

2.5 Solutions to the three problems

The solutions to basic three problems are mathematically linked to each other under probabilistic frame work.

2.5.1 Problem 1 -Probability evaluation

It is the problem of computing the probability of observation sequence $O = (o_1, o_2 \ldots . o_T)$, given the model λ i.e $P(O|\lambda)$. Consider the state sequence as

$$q = (q_1, q_2, \ldots \ldots q_T), \tag{2.16}$$

where q_1 is the initial state. The probability of the observation sequence O, given the state sequences q, is computed as

$$P(O|q, \lambda) = \prod_{t=1}^{T} P(o_t|q_t, \lambda) \tag{2.17}$$

here the observation sequences is assumed to be statistically independent . Thus we get

$$P(O|q,\ \lambda) = b_{q_1}(o_1).b_{q_2}(o_2).\ldots.b_{q_T}(o_T) \tag{2.18}$$

The probability of such a state sequence can be written as

$$P(q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3}.\ldots.a_{q_{T-1} q_T}. \tag{2.19}$$

The joint probability of O and q, i.e the probability that O and q occur simultaneously is written as

$$P(O, q|\lambda) = P(O|q, \lambda)P(q|\lambda). \tag{2.20}$$

The probability of observation sequence O, given the model, is obtained by summing this joint probability over all possible state sequences q , giving

$$\begin{aligned} P(O|\lambda) &= \sum_{all\ q} P(O|q, \lambda)P(q|\lambda) \\ &= \sum_{q_1 q_2 .. q_T} \pi_{q_1} b_{q_1}(o_1) a_{q_1 q_2} b_{q_2}(o_2) \ldots a_{q_{T-1} q_T} b_{q_T}(o_T) \end{aligned} \tag{2.21}$$

The calculation of $P(O|\lambda)$, according to the direct definition given in 2.21 involves on the order of $2 * T * N^T$.Therefore a more efficient calculation procedure is required to solve the problem -1. This procedure is called *Forward Procedure*

The forward Procedure

Consider the forward variable $\alpha_t(i)$ defined as

$$\alpha_t(i) = P(o_1, o_2, o_3.\ldots.o_t, q_t = i|\lambda) \tag{2.22}$$

i.e., the probability of the partial observation sequence $o_1 o_2 \ldots o_t$ (until time t) and state i at time t, given the model λ.. $\alpha_t(i)$ can be solved inductively as

1. Initialization

$$\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \leq i \leq N \tag{2.23}$$

2. Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(o_{t+1}), \quad \begin{matrix} 1 \leq t \leq T-1 \\ 1 \leq j \leq N \end{matrix} \tag{2.24}$$

3. Termination

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i). \tag{2.25}$$

Step 1 initializes the forward probabilities as the joint probability of state i and initial observation o_t. Step 2 accounts for how state j is reached at time $t+1$ from the N possible states i, $i = 1, 2, \ldots N$. at time t. Step 3 gives the desired calculation of $P(O|\lambda)$ as the sum of the terminal forward variables $\alpha_T(i)$. The number of calculations required in this procedure is on the order of $N^2 * T$.

The Backward Procedure

In a similar manner we can consider a backward variable $\beta_t(i)$ defined as

$$\beta_t(i) = P(o_{t+1}, o_{t+2} \quad o_T | q_t = i, \lambda) \tag{2.26}$$

i.e., the probability of the partial observation sequence from $t+1$ to the end, given the state i at time t and the model λ. Again $\beta_t(i)$ cab be solved inductively as follows

1. Initialization

$$\beta_T(i) = 1, \qquad 1 \leq i \leq N. \tag{2.27}$$

2. Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \quad \begin{matrix} 1 \le i \le N \\ t = T-1, T-2, \ldots 1 \end{matrix} \tag{2.28}$$

The initialization step arbitrarily defines $\beta_T(i)$ to be 1 for all i .Step 2 shows that in order to be in state i at time t, and to account for the observation sequence from time $t+1$ on, we have to consider all possible states j at time $t+1$, accounting for the transition from i to j (the term a_{ij}), as well as the observation o_{t+1} in state j (the $b_j(o_{t+1})$ term), and then account for the remaining partial observation from state j (the $\beta_{t+1}(j)$ term). The number of calculations required in this procedure is on the order of $N^2 * T$ only.

2.5.2 Problem 2 -Optimal state sequence

There are several ways of solving the problem 2 i.e., to find the optimal sequence associated with the given observation sequence . A formal technique for finding the single best sequence is based upon dynamic programming method called *Viterbi algorithm.*

Viterbi Algorithm (VA)

To find the single best state sequence $q = (q_1, q_2, \ldots q_T)$, for the given observation sequence $O = (o_1 o_2 \ldots . o_T)$, we define the quantity

$$\delta_t(i) = \max_{q_1, q_2, \ldots q_T} P(q_1 q_2 \ldots q_{t-1}, q_t = i, o_1 o_2 \ldots o_t | \lambda). \tag{2.29}$$

i.e., $\delta_t(i)$ is the best sequence along a single path at time t, which accounts for the first t observation sequences and ends in state i. The complete procedure for finding the sequence is given below

1. Initialization

$$\delta_1(i) = \pi_i b_i(o_1), \quad 1 \leq i \leq N \tag{2.30}$$

2. Recursion

$$\delta_t(j) = \max_{1 \leq i \leq N} [\delta_{t-1}(i) a_{ij}]\, b_j(o_t), \quad \substack{2 \leq t \leq T \\ 1 \leq j \leq N} \tag{2.31}$$

3. Termination

$$P^* = \max_{1 \leq i \leq N} [\delta_T(i)] \tag{2.32}$$

In the viterbi algorithm, the computation of $\delta_t(i)$ involves a and b terms which are less than one. The value of $\delta_t(i)$ gets smaller and smaller as the time starts to grow big. Hence an alternative procedure is proposed.

Alternative viterbi implementation

It uses the logarithmic value of model parameters . Following steps are used

1. Pre-procession

$$\tilde{\pi}_{i=} \ln(\pi_i) \qquad 1 \leq i \leq N \tag{2.33}$$

$$\tilde{b}_i(o_t) = \ln[b_i(o_t)], \qquad 1 \leq i \leq N,\ 1 \leq t \leq T \tag{2.34}$$

$$\tilde{a}_{ij} = \ln(a_{ij}), \qquad 1 \leq i,\ j \leq N \tag{2.35}$$

2. Initialization

$$\tilde{\delta}_1(i) = \ln(\delta_1(i)) = \tilde{\pi}_i + \tilde{b}_i(o_1), \quad 1 \leq i \leq N \tag{2.36}$$

3. Recursion

$$\delta_t(i) = \ln(\delta_1(i)) \max_{1 \le i \le N}[\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}] + \tilde{b}_j(o_t) \qquad 2 \le t \le T, \quad 1 \le j \le N \tag{2.37}$$

4. Termination

$$\tilde{P}^* = \max_{1 \le i \le N}[\tilde{\delta}_T(i)] \tag{2.38}$$

The calculations required for this alternative implementation are on the order of N^2T additions. Because the pre-processing needs are to be performed once and saved, its cost is negligible for most system.

2.5.3 Problem 3-Parameter Estimation

It is the most difficult problem of hidden Markov model. There is no known way to analytically solve for the model parameter set that maximizes the probability of the observation sequence in closed form. However model $\lambda = (A, B, \pi)$ can be chosen such that its likelihood, $P(O|\lambda)$, is locally maximized using the iterative procedure such as the *Baum-Welch* method (also known as expectation-maximization method). In case of speech processing this is often called training and the given observation sequence based on which we obtain the model parameters is called training sequence.

Baum-Welch iterative procedure

Let define $\xi_t(i, j)$as

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j|O, \lambda), \tag{2.39}$$

i.e., the probability of a path being in state i at time t and making a transition to state j at time $t+1$, given the observation sequence and the model. $\xi_t(i,j)$ can be written as

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)} \tag{2.40}$$

Let us define $\gamma_t(i)$ as the probability of state i at time t, given the entire observation sequence and the model, Mathematically

$$\gamma_t(i) = \sum_{j=1}^{N}\xi_t\ (i,j)\,, \tag{2.41}$$

Hence the re-estimation formulae for A, B, π in term of these parameters are given as

$$\begin{aligned}\pi &= \textit{expected frquency (number of time) in state } i \\ &\qquad \textit{at time } (t=1) = \gamma_1(i)\end{aligned} \tag{2.42}$$

$$\begin{aligned}a_{ij} &= \frac{\textit{expected number of transition from state } i \textit{ to state } j}{\textit{expected number of transition from state } i} \\ &= \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)}\end{aligned} \tag{2.43}$$

$$\begin{aligned}b_j(k) &= \frac{\textit{expected number of times in state } j \textit{ and observing symbol } v_k}{\textit{expected number of time in state } j} \\ &= \frac{\sum_{t\ni o_t=v_k}^{T-1}\gamma_t(j)}{\sum_{t=1}^{T-1}\gamma_t(j)}\end{aligned} \tag{2.44}$$

The re-estimation formula can be derived by maximization of Baum's auxiliary function defined as

$$Q(\lambda', \lambda) = \sum_{q} P(O, q|\lambda') log P(O, q|\lambda), \tag{2.45}$$

over λ. Because

$$Q(\lambda', \lambda) \geq Q(\lambda', \lambda') \Rightarrow P(O|\lambda) \geq P(O|\lambda') \tag{2.46}$$

we can maximize the function $Q(\lambda', \lambda)$ over λ to improve λ' in the sense of increasing the likelihood $P(O|\lambda)$. Eventually the likelihood function converges to a critical point if we iterate the procedure.

The re-estimation formula is hence represented as

$$\tilde{\pi} = \frac{\alpha_1(i)\beta_1(i)}{\sum_{j=1}^{N} \alpha_T(j)} = \gamma_1(i) \tag{2.47}$$

$$\tilde{a}_{ij} = \frac{\sum_{t=1}^{T} \alpha_t(i)\, a_{ij}\, b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{t=1}^{T-1} \alpha_t(i)\beta_t(i)} \tag{2.48}$$

$$\tilde{b}_j(k) = \frac{\sum_{t \ni o_t = v_k}^{T} \alpha_t(i)\beta_t(j)}{\sum_{t=1}^{T} \alpha_t(j)\beta_t(j)} \tag{2.49}$$

Remark

Successful application of Hidden Markov Model methods usually involves the following steps:

1. define the set of M sound classes for modeling, such as phonemes or words.

These sound classes are represented as V=$\{v_1\, v_2 \ldots . v_L\}$.

2. for each class, collect a sizable set (the training set) of labeled utterances that are known to be in that class.

3. based on each training set solve the estimation problem to obtain the best model λ_i for each class v_i $i = 1, 2, ..L$.

4. during recognition, evaluate $P(O|\lambda_i), i = 1.2...L$, for the unknown utterance sequence and identify the speech that produces observation sequence O as class v_j if

$$P(O|\lambda_j) = \max_{1 \leq i \leq L} P(O|\lambda_i).$$

Here We have introduced briefly the steps taken during the implementation of HMM. The detailed procedure is discussed in next chapter.

Chapter 3

Implementation of HMM for isolated word recognition

In the previous chapter probabilistic framework of Hidden Markov Model has been discussed in detail. The complete HMM model is represented as $\lambda = (A, B, \pi)$. The output distribution $\{b_{ik}\}$ models the parametric distribution of speech event and the transition distribution $\{a_{ij}\}$models the duration of these events. How the concept of HMM is applied to isolated word recognition system, is discussed in this chapter. Initially the strategy of processing is explained theoretically and then the process of implementation of HMM is explained.

3.1 Strategy Of Processing

The speech signal is slowly time varying signal in the sense that, when examined over a sufficiently short duration of time, its characteristics are fairly stationary. Therefore it can be represented by linear time invariant models such as autoregressive (AR), linear prediction, the short time Fourier transform (STFT) or wavelet transform (WT). In this work linear prediction model is

considered. LPC provides a good model of the speech signal and works well in recognition. However, for the sequence of long duration corresponding to a word (about some hundred millisecond to one second), the speech signal is a time-variant signal in the amplitude and in frequency components. The speech signal corresponding to the whole duration of word is divided into several segments each of several tens of milliseconds. Each such segment is called a frame. Each frame is represented by the characteristic vector called observation vector O_t . Here, the cepstral coefficient is used to represent a frame.

For the discrete HMM we have to design a codebook consisting of finite distinct vectors (symbols). Inputs to the isolated word recognition system based on discrete HMM, are sequences of discrete symbols chosen from codebook. To determine the entries of codebook we proceed as follows. The vector corresponding to each frame of a word is obtained. These vectors are grouped into a set. Similarly we determine a set of vectors for each word in vocabulary. These sets of vectors are combined to give a set of training vectors. Let a set $\{o_i \forall i = 1, 2, .. \text{ I}\}$ denotes the training set of vectors that occurred when the words in vocabulary are pronounced. By K-mean clustering algorithm we partitioned the vectors, o_i, into M different cells. Then centroid of each cell is so computed such that the average distortion in replacing each of the training set vectors o_i by the closest centroid is minimum. Each centroid so obtained is registered in our codebook. Once a codebook is deigned, the mapping between continuous vectors and codebook indices is done by nearest distance rule.

Hidden Markov model is used to model the transition between the frames which are non-stationary processes. During training phase, different HMMs are defined for different words in vocabulary. In this work six HMMs are defined. During recognition, *Viterbi algorithum*[8, 7] is used to determine the HMM within the set of HMMs that best matches the observation sequence by means

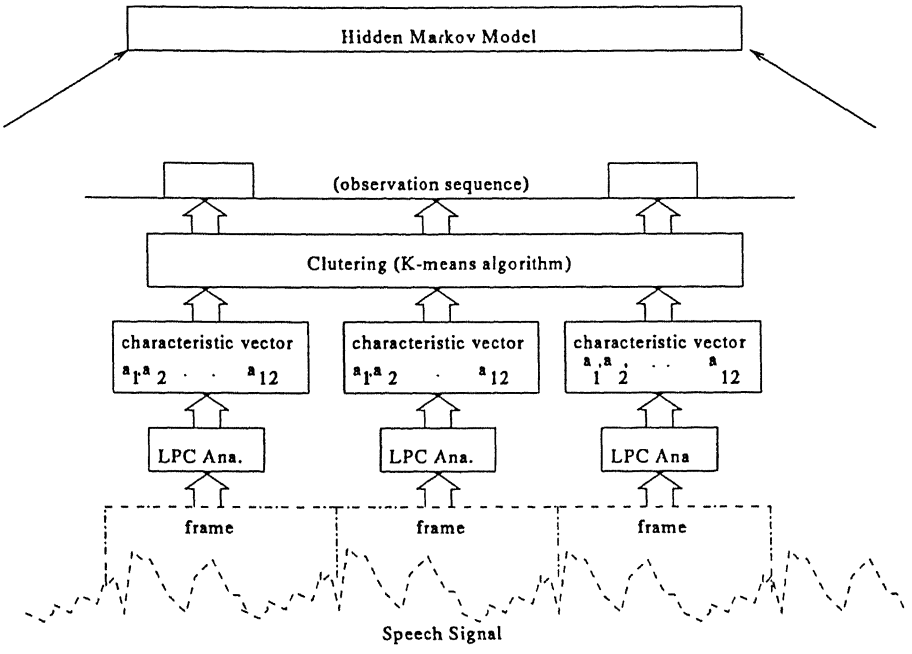


Figure 3.1: The strategy of processing

of computing the likelihood score of the most likely state sequence in each HMM and selects the HMM with the highest likelihood score to be the HMM that best matches with this observation.

The strategy of processing is illustrated in fig 3.1. The steps used are following.

- Front end processing,
- Clustering,
- Training of HMM,
- Recognition

These steps are elaborated in next sections.

The speech signal contains many frequency components. It becomes necessary to detect the speech in presence of noise. By accurately detecting the

beginning and end of the utterance, the speech data can be kept at minimum. This problem is solved by the *End point detector.*

3.2 End point detector

The objective of speech detection is to separate acoustic events of interest in a continuously recorded signal from other part of signal. The end point detector[14, 15] must have following features.

- Simple and efficient processing.
- Reliable location of significant acoustic events.
- Capability of being applied to varying background silences.

The algorithm proposed for locating the endpoints of an utterance is based on two measures of signal, zero crossing rate and energy. The speech "energy" is defined as the sum of the magnitudes of 10ms of speech centered on the measurement interval, i.e.

$$E(n) = \sum_{i=-50}^{50} |s(n+i)|, \tag{3.1}$$

where s(n) are the speech samples and it is assumed that sampling frequency is $10kHz$. The following steps are used in this algorithm.

1. Obtain the maximum energy E_{max} for the silent portion.
2. Compare each frame (10ms duration) of recorded data with E_{max} . If energy of frame is greater than E_{max} then label that point as the starting point of that word.

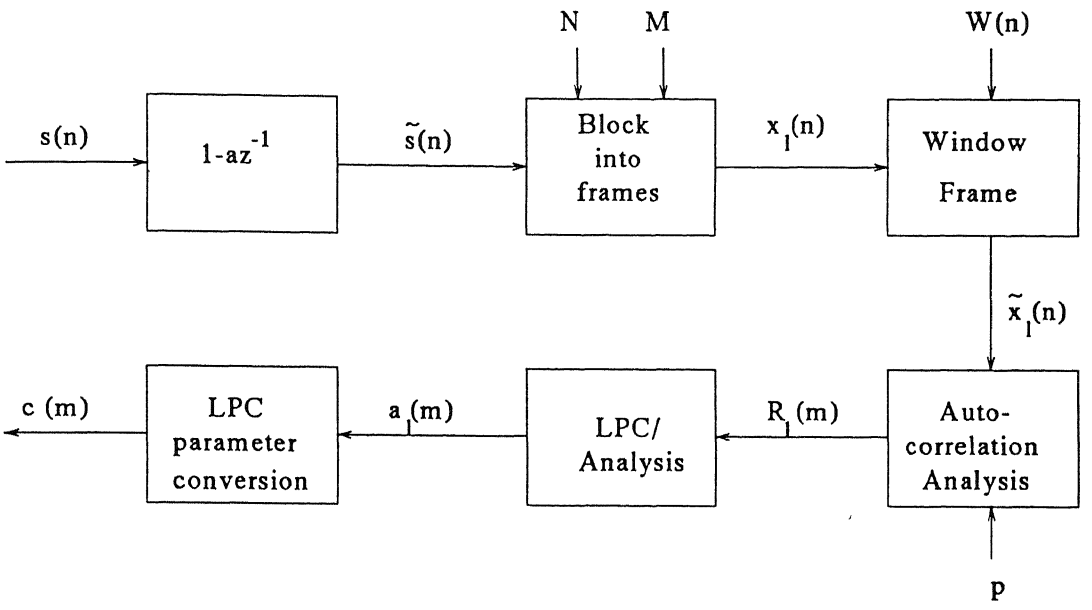


Figure 3.2: Front-end processor

3. Now go on comparing energy of each frame with E_{max} , till E_{max} is less than energy of the frame. When E_{max} is greater then label that point as end point.

By means of end point algorithm all background noise after the word or before the word are removed to acceptable label. After the end point detector, data is available for processing further i.e for determining observation vector.

3.3 LPC Feature Analysis

LPC[1] based front-end processor is used in this work for the implementation of speech-recognition system. Fig 3.2 shows the block diagram of LPC processor[2, 4].

The above system is a block processing model in which a frame of N samples is processed and a vector of features O_t is computed. Following steps are

[1]see Appendix B

involved in the processing:

1. Pre-emphasis- The speech signal is digitized at sampling rate of $10kHz$ and processed by a first order digital network in order to spectrally flatten the signal. Most widely used pre-emphasis is

$$H(z) = 1 - az^{-1} \qquad 0.9 \leq a \leq 1.0 \tag{3.2}$$

In this case, output $\tilde{s}(n)$, is related to the input to the network, $s(n)$, by the difference equation

$$\tilde{s}(n) = s(n) - a * s(n-1). \tag{3.3}$$

Here the value of a is taken to be 0.95.

2. Frame Blocking- In this step the pre-emphasized speech signal, $\tilde{s}(n)$, is blocked into frames of N samples, with adjacent frames being separated by M samples. Here N is taken as 256 and consecutive frames are separated by the 85 samples. If $M \leq N$ then adjacent frames overlap and the resulting LPC spectral estimates will be correlated from frame to frame, if $M \ll N$ then LPC spectral estimates from frame to frame will be quite smooth . On the other hand if $M > N$, then there will be no overlapping between adjacent frames. Let l^{th} frame of speech is denoted by $x_l(n)$, and there are L frames within the entire speech signal, then

$$x_l(n) = \tilde{s}(M * l + n), \qquad n = 0, 1, .. N-1, \;\; l = 0, 1...L-1 \tag{3.4}$$

3. Frame Windowing- Each frame is windowed i.e multiplied by window so as to minimize the signal discontinuities at the beginning and end of each

frame. By this means the signal is being tapered to zero at the beginning and end of each frame. Let $w(n)$, $0 \leq n \leq N-1$ denotes the window then the result of windowing is the signal

$$\tilde{x}_l(n) = x_l(n)w(n), \qquad 0 \leq n \leq N-1 \tag{3.5}$$

Here *Hamming Window* is used. It is of the form

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right), \qquad 0 \leq n \leq N-1 \tag{3.6}$$

4. Autocorrelation Analysis- Each frame of windowed signal is next autocorrelated to give

$$r_l(m) = \sum_{n=0}^{N-1-m} \tilde{x}_l(n) * \tilde{x}_l(n+m), \qquad m = 0, 1, \ldots p \tag{3.7}$$

where the highest autocorrelation value, p, is the order of the LPC analysis. Here the value of p is taken as 10.

5. LPC Analysis- For each frame, a vector of LPC coefficient is computed. The formal method for converting from autocorrelation coefficients to an LPC parameter set (for LPC autocorrelation method) is known as *Durbin's method*[2]. An LPC derived cepstral vector is then computed up to the Q^{th} component, where $Q > P$ and $Q = 12$ is used here.

6. LPC Parameter Conversion to Cepstral Coefficients- A very important LPC parameter set, which can be derived directly from the LPC coefficient set, is the LPC cepstral coefficient, $c(m)$. The recursion used is

$$c_0 = \ln \sigma^2 \tag{3.8}$$

$$c_m = a_m + \sum_{k=1}^{m-1} \left(\frac{k}{m}\right) c_k a_{m-k}, \qquad 1 \leq m \leq p \tag{3.9}$$

$$c_m = \sum \left(\frac{k}{m}\right) c_k a_{m-k} \qquad m > p, \tag{3.10}$$

where σ^2 is the gain term in the LPC model. The cepstral coefficients are more reliable feature sets for speech recognition than the LPC coefficients. Generally the value of Q is taken as 12.

Complete LPC analysis is shown in fig (3.2). After performing LPC analysis frame by frame, we obtain a set of observation vectors. Each vector has a dimension of 12. (In this work we have obtained 23 vectors, each of dimension 12, for a vowel). Similarly we obtained another set of 23 vectors for another vowel. Total no of vectors obtained are $23 \times 6 = 138$. These vectors are grouped into a set of 6 distinct vectors.

3.4 K-Mean Clustering

The purpose of vector quantization is to design a codebook containing distinct finite set of vectors. After the LPC analysis we obtain a set $V = \{v_1, v_2, \ldots . v_L\}$ containing L number of training vectors. We have obtained total 138 vectors. We want to classify these vectors into M number of different groups (or cells) and to determine a centroid c_j for each group (or cell) C_j representing that cell. The centroid c_j must be such that, average distortion in cell C_j is minimum. i.e to $minimize\, E[d(v_i^j, c_j)|v_i^j \in C_j]$ where v_i^j denotes the vector belonging to cell C_j.

With the help of K-mean algorithm[10, 16] we determine c_j for each C_j by

[2]see appendix B

minimizing the average distortion defined by

$$D = \frac{1}{L}\sum_{i=1}^{L} d(v_i, c_{v_i^j}) \tag{3.11}$$

where v_i denotes a vector from set V and $c_{v_i^j}$ denotes the centroid of the cell C_j to which vector v_i belongs. The distortion measure used here is

$$d(v_i, c_{v_i^j}) = (v_i - c_{v_i^j})^T (v_i - c_{v_i^j}). \tag{3.12}$$

The centroid of the cell C_j is calculated as :

$$c_j = \frac{1}{N_j}\sum_{n=1}^{N_j} v_n^j \tag{3.13}$$

where N_j is total number of vectors in cell C_j.

The procedure is described in following steps.

1. Initially partition the set of L vectors into M different cells arbitrarily. Determine the centroid c_i as

$$c_i = \frac{1}{N_j}\sum_{n=1}^{N_j} v_n^j \qquad 1 \le j \le M \tag{3.14}$$

2. Proceed through the set of L vectors and assign the vector v_i to the cell C_j whose centroid c_j is the nearest to it. Mathematically calculate

$$d_j = d(v_i, c_j) \qquad 1 \le j \le M \tag{3.15}$$

and transfer the v_i to cell C_j for which d_j is minimum.

3. Recalculate the centroid for the cell receiving the new vector and for the cell losing the vector.

4. Repeat step 2 and 3 until no more reassignments take place.

In this way a codebook of desired size is obtained. Then during training and recognition the continuous vectors are assigned to the index of nearest codebook vector.

3.5 Block diagram of isolated word recognition

Consider the block diagram (fig. 3.3) for understanding the concept of isolated word recognition system. The word for which a HMM is to be designed, is uttered repeatedly and the training set of LPC derived cepstral vector is obtained by front end processor. These observation vectors are compared one by one with each vector in codebook and are assigned the index of the nearest codebook vector. By this way we obtain the observation sequence (index sequence) $O = \{o_1, o_2, \ldots o_T\}$ which is used in training of HMM. During training $Baum Welch$ re-estimation algorithm is used. The starting conditions used for the re-estimation algorithm are

$a_{ij} = 1/N \qquad \forall\, N = total\ number\ of\ states$

$b_{jk} = 1/M \qquad \forall\, M = total\ number\ of\ vectors\ in\ codebook.$

Hence after training we obtain a HMM representing that word. Similarly we design different HMMs representing different words.

During recognition, switch is changed to mode 2. We determine observation sequence for the test word and the probability score is calculated using viterbi algorithm.

The complete isolated word recognition model is shown in fig 3.4. We have a vocabulary of six words to be recognized and each word is modeled by a distinct HMM. For each word in the vocabulary we have a training set of K occurrences of each spoken word where each occurrence of the word constitutes

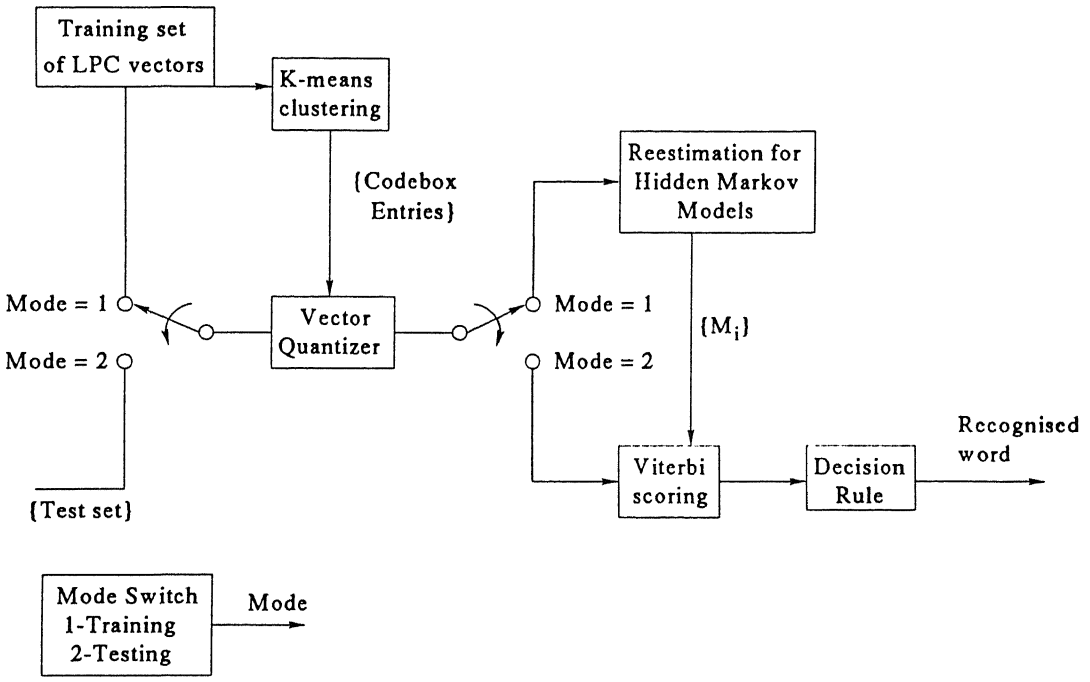


Figure 3.3: Conceptual Model for isolated word recognition

an observation sequence. The observations are some appropriate representation of the characteristics of the word. In order to do isolated word recognition, we must perform the following:

1. For each word v in the vocabulary, we must build an HMM λ^v i.e we must estimate the modal parameters (A, B, λ) that optimize the likelihood of the training set observation vectors of the vth word.

2. For each unknown word which is to be recognized, the processing described above is carried out and probability is calculated

$$v = argmax_{1 \leq v \leq V} [P(O|\lambda^v] \tag{3.16}$$

for each model and HMM is selected accordingly.

The probability calculation is performed using the Viterbi algorithm and it requires on an order of $V * N^2 * T$ computation.

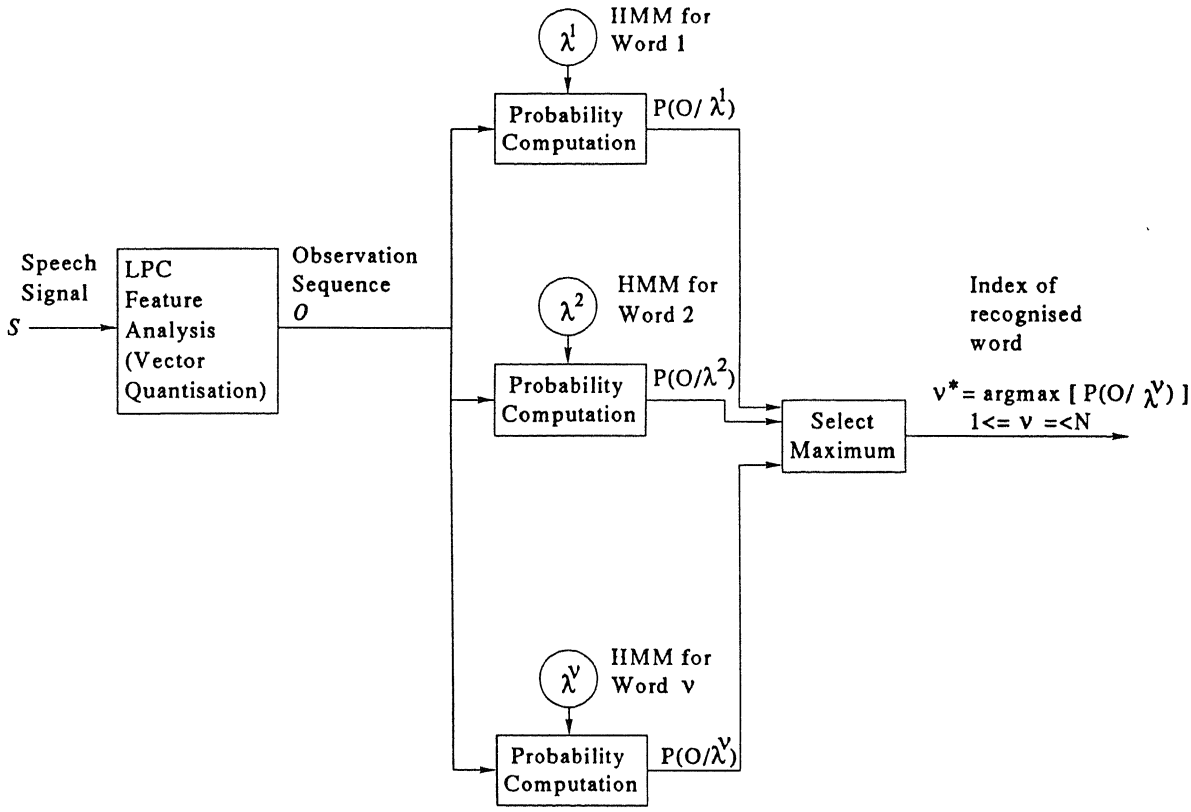


Figure 3.4: The complete model for isolated word recognition

3.6 Applying the VA to HMMs

Given the observation sequence, the VA finds the most likely state sequence in a given HMM and the likelihood associated with this most likely sequence. At the beginning of each search, each state in a given HMM has some predefined likelihood score. To find the most likely transition and update the state likelihood score for each state at any time instant, one must find the most likely transition coming into a given state. This is done by adding the transition probabilities of all the transitions coming into this given state to their corresponding previous state likelihood score, and selecting the transition with maximum sum to be the most likely transition coming into this state This sum is then added to the observation probability assigned by the given state, to current observation symbol, in order to form the updated likelihood score for this given state. The most likely transition and the state likelihood score are updated for all states in recursion. This process is performed recursively until all symbols in the given observation sequence are processed. At the end of recursion, the path associated with the state, that has the highest likelihood score, is selected as the most likely state sequence for the given observation. That is, if the observation sequence is $(o_1, o_2 \ldots o_k)$, then for state j at recursion t, one wants to compute

$$\delta_t(t) = max_{1 \leq i \leq N} \{\delta_{t-1}(i) + \ln(a_{ij}) + \ln(b_j(o_t))\} \tag{3.17}$$

where $b_j(o_t)$ is the observation probability for o_t assigned by state j at recursion t, a_{ij} is the transition probability of the transition from the state i to state j, δ_{ij} is the state likelihood score for the state j at a recursion t.

At the end the of recursion, the path that has the highest likelihood score is selected to be the most likely path or state sequence. For determining the most likely state sequence, the most likely transition for each state is registered

at each recursion so that the most likely state sequence can be traced at the end of the recursion.

Applying viterbi to calculate the likelihood score provides the following advantages. Viterbi based search is more efficient. If we use a logarithmic representation of HMM probabilities, viterbi search becomes extremely efficient because the multiplication is reduced to addition. It is also possible to obtain state sequence with viterbi algorithm.

Chapter 4

Results And Discussion

Procedure for the implementation of speech recognition system based on Hidden Markov model is explained in previous chapter. System has been designed for the recognition of six vowels. These vowels are - ϵ (bet), $\wedge$ (but), a (hot), I (bit), O (obey) and U (food). Six discrete HMMs are trained for these six vowels each representing one vowel.

These vowels are recorded in the noise free environment using a sampling rate of 10kHz. Speech signals for vowels are shown in fig(4.1- 4.6). LPC is used for determining the spectral properties of the signal. According to this algorithm the speech samples are first pre-emphasized by using first order filter $1-0.95z^{-1}$. This pre-emphasized sample is then blocked into numbers of small frame. Frame size is taken as 256 samples. Consecutive frames are spaced 85 samples apart. There is overlapping of 171 samples. Each frame is windowed by using Hamming Window given as

$$w(n) = 0.54 - 0.46 * cos(2\pi n/(N-1)), \quad where\, 0 \leq n \leq N-1 \qquad (4.1)$$

LPC coefficients are computed for each frame and then it is converted into

cepstral coefficient. All the vectors obtained after front end processing are clustered into six cells by using K-mean algorithm and centroid of each cell is computed. Table (4.1) shows the centroid of each cell. The model parameters are computed by using Baum-Welch algorithm. Starting conditions used in re-estimation algorithm are:

1. $\pi_i = 1/N \qquad 1 \leq i \leq N;$
2. $a_{ij} = 1/N \qquad 1 \leq i, j \leq N;$
3. $b_{ik} = 1/M \qquad 1 \leq i \leq N, 1 \leq k \leq M.$

Following constraints are applied on b_{ik}

1. if

   $b_{ik} \leq 1 * 10^{-5}$;

   then

   $b_{ik} = 1 * 10^{-5}$;

2. $\sum_k b_{ik} = 1;$

Results obtained are given in tables (4.2-4.7).

While determining left-right HMM (discussed in chapter 2) parameters, following constraints are applied on state transition matrix and on initial transition matrix in addition to above constraints.

1. $a_{ij} = 0 \qquad j > i;$
2. $\pi_i = \begin{cases} 0 & i \neq 1 \\ 1 & i = 1 \end{cases}$

Calculated values of parameters are given in tables (4.8-4.14)

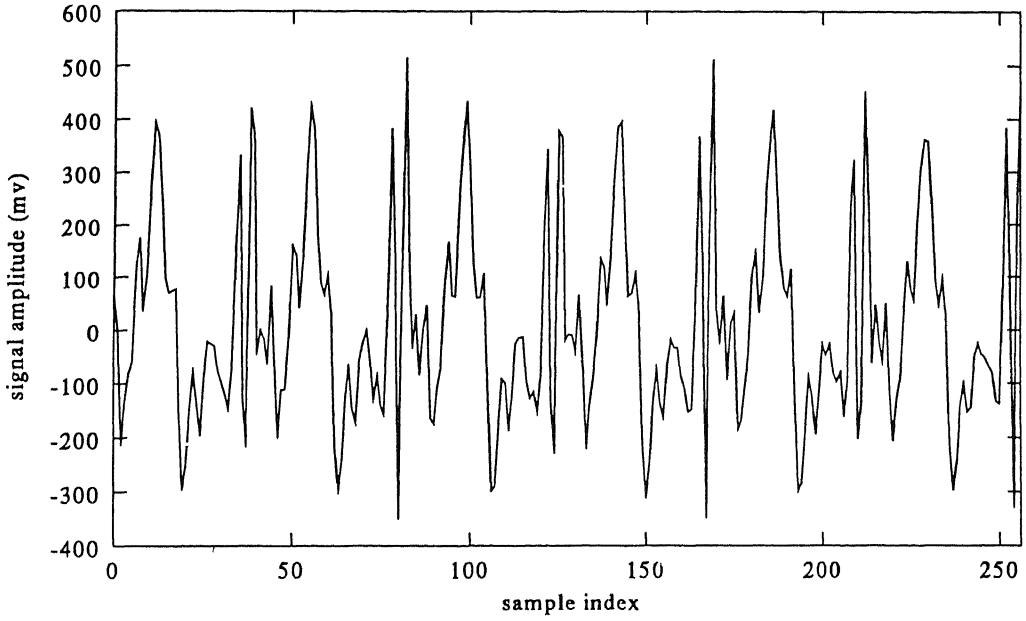


Figure 4.1: Speech signal for utterance of vowel - ϵ (bet)

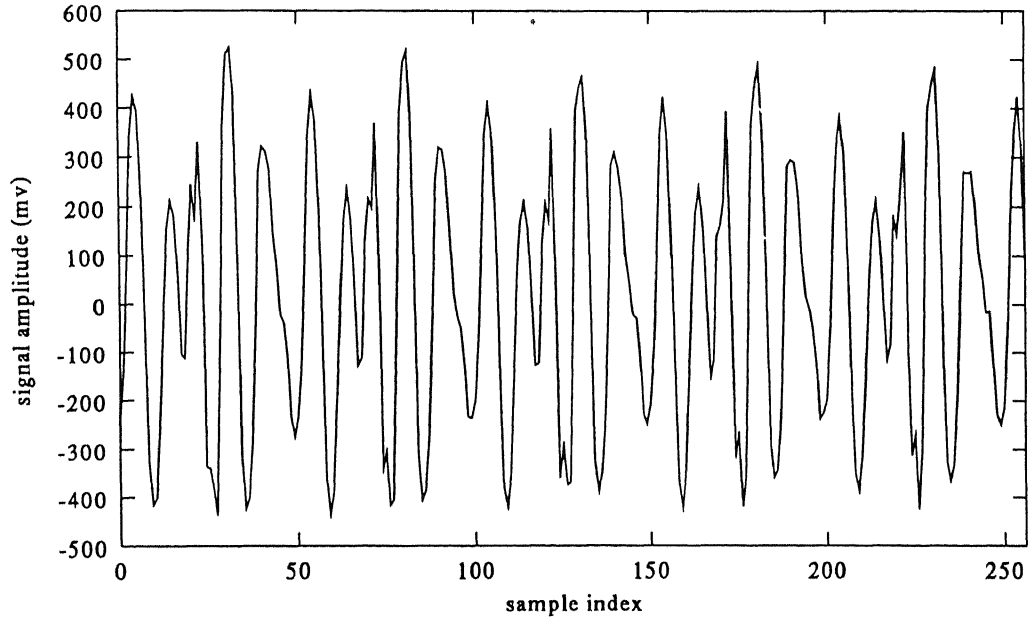


Figure 4.2: Speech signal for utterance of vowel - $\wedge$ (but)

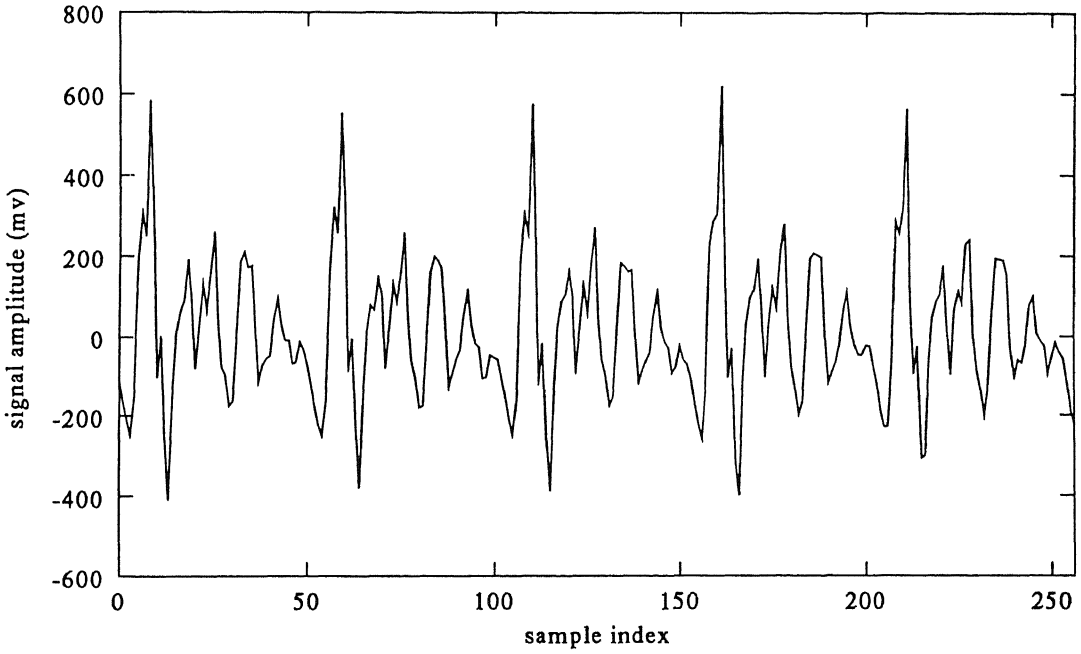


Figure 4.3: Speech signal for utterance of vowel - a (hot)

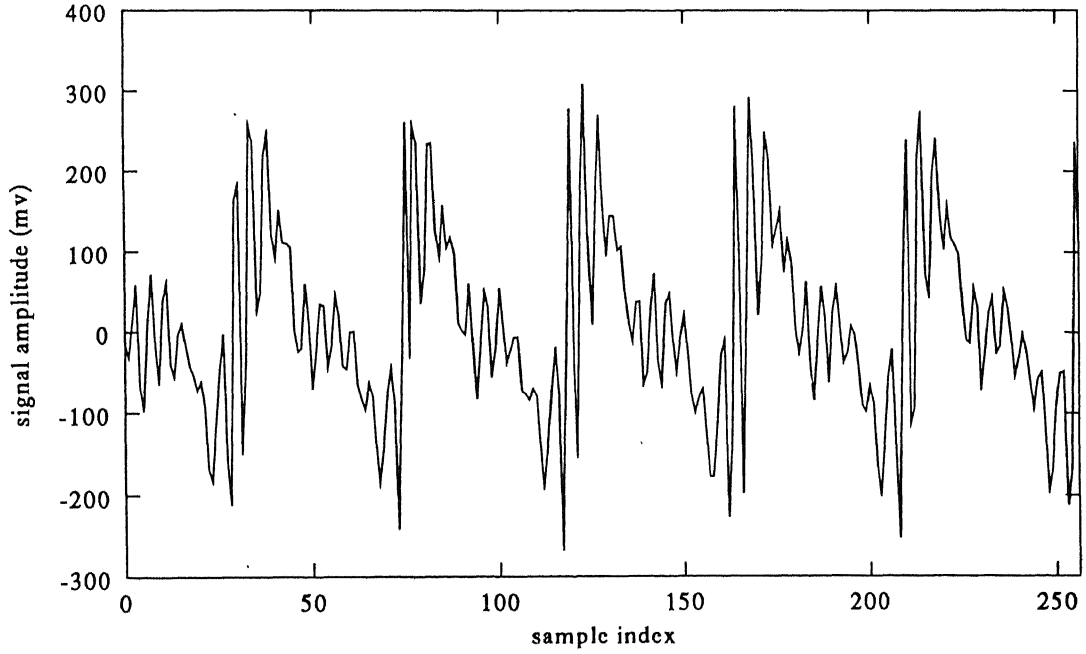


Figure 4.4: Speech signal for vowel - I (bit)

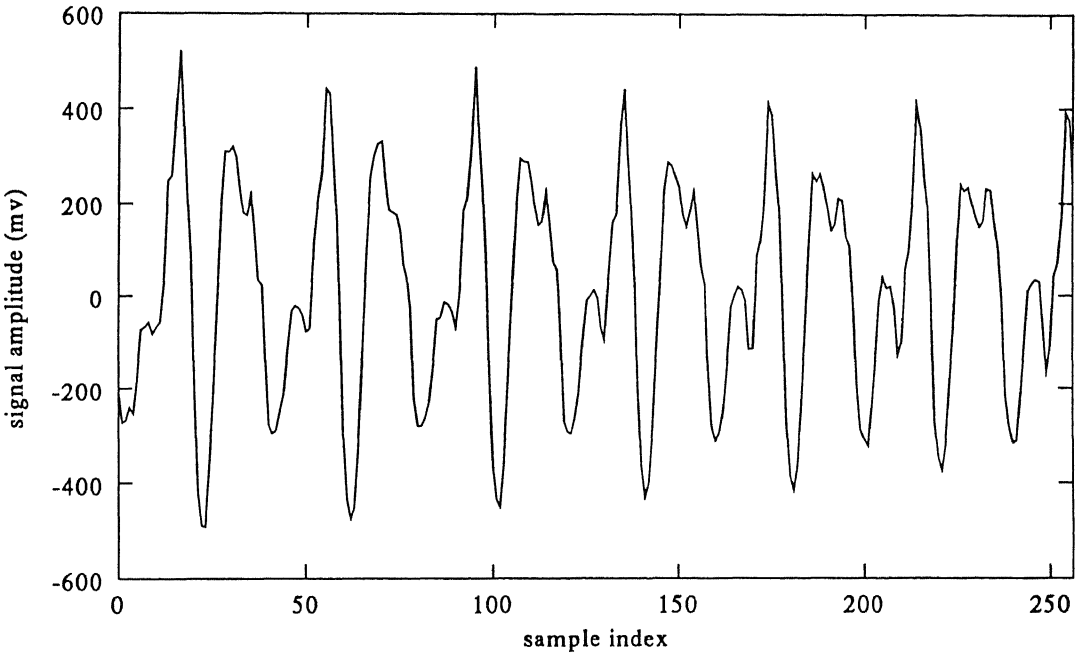


Figure 4.5: Speech signal for vowel - O (obey)

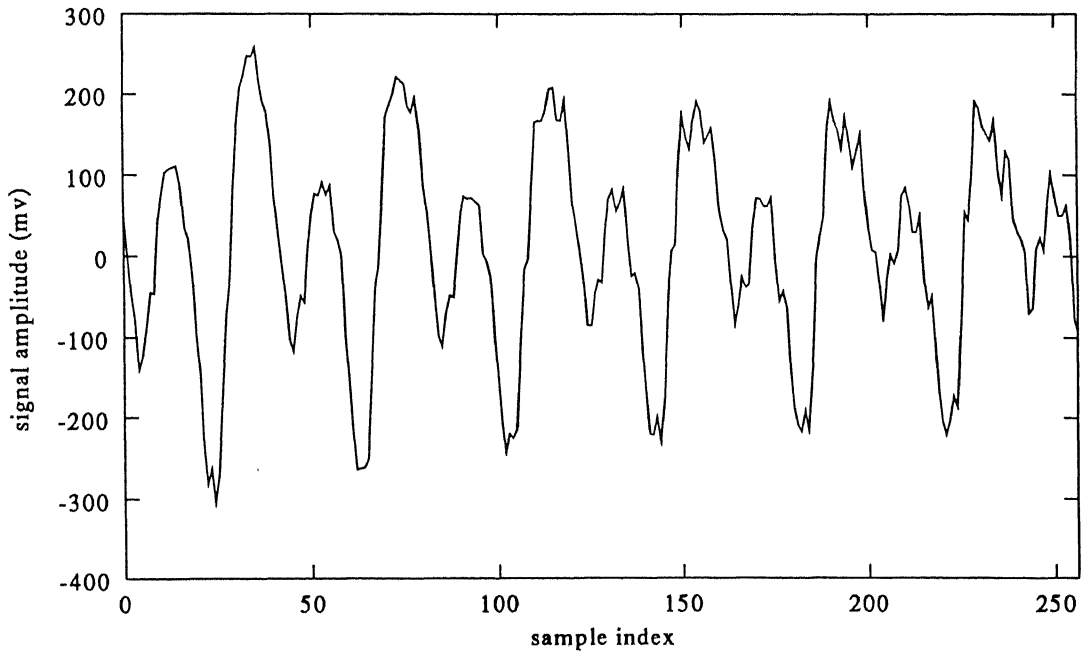


Figure 4.6: Speech signal for vowel - U (food)

cell no	centroid (cepstral coeff. values) of cell			
1	0.831882	−4.642482	3.991625	−4.065114
	4.834633	−2.659011	0.830542	−3.114954
	−0.816965	−1.441938	−1.811512	0.605486
2	0.756187	−1.161783	2.945668	−6.533916
	−3.190974	−2.963116	2.925372	−1.242985
	−1.590204	−1.624079	−1.095897	0.04612
3	0.194863	−0.6100089	3.977964	−5.893832
	−3.19.974	−4.490866	4.556600	−0.298862
	−1.108746	−2.089710	−0.991159	−0.18204
4	−0.50897	−3.796737	2.486141	1.650537
	6.558616	1.119739	0.634291	−3.690248
	−1.753764	−1.398450	0.706641	0.072938
5	1.143870	0.707063	3.137482	−3.736169
	−3.861398	−2.343838	3.506834	1.775789
	−0.145828	−1.925398	−0.467580	−0.571358
6	0.883796	1.288458	2.260686	−0.589854
	−3.246550	−1.579590	−0.713658	2.495806
	0.431527	−1.523267	0.443054	−0.019359

Table 4.1: Output of clustering algorithm

$\pi_i =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$a_{ij} =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$b_{ik} =$

$0.99995e-01$	$0.99995e-01$	$0.99995e-01$	$0.99995e-01$	$0.99995e-01$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.2: Reestimation model parameters for vowel "ϵ (bet)"

$\pi_i =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$a_{ij} =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$b_{ik} =$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$0.99995e-0$□	$0.99995e-0$□	$0.99995e-0$□	$0.99995e-0$□	$0.99995e-0$□
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.3: Reestimation model parameters for vowel "$\wedge$ (but)"

$\pi_i =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$a_{ij} =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$b_{ik} =$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$0.99995e-01$	$0.99995e-01$	$0.99995e-01$	$0.99995e-01$	$0.99995e-01$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.4: Reestimation model parameters for vowel "a (hot)"

$\pi_i =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$a_{ij} =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$b_{ik} =$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$0.99995e-0$	$0.99995e-0$	$0.99995e-0$	$0.99995e-0$	$0.99995e-0$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.5: Reestimation model parameters for vowel "I (bit)"

$\pi_i =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$a_{ij} =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$b_{ik} =$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$0.99995e-0\mathfrak{D}$	$0.99995e-0\mathfrak{D}$	$0.99995e-0\mathfrak{D}$	$0.99995e-0\mathfrak{D}$	$0.99995e-0\mathfrak{D}$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.6: Reestimation model parameters for vowel "O (obey)"

$\pi_i =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$a_{ij} =$

$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$
$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$	$2.00000e-01$

$b_{ik} =$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$0.99995e-01$	$0.99995e-01$	$0.99995e-01$	$0.99995e-01$	$0.99995e-01$

Table 4.7: Reestimation model parameters for vowel "U (food)"

$\pi_i =$

$1.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$

$a_{ij} =$

$9.54210e-01$	$2.61651e-02$	$9.81194-e02$	$5.72363-e-03$	$4.08830e-03$
$0.00000e-00$	$5.71423e-01$	$2.14882e-01$	$1.25001e-01$	$8.92867e-02$
$0.00000e-00$	$0.00000e-00$	$4.99993e-01$	$2.91666e-01$	$2.08333e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$5.83333e-01$	$4.16667e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$1.00000e-00$

$b_{ik} =$

$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.8: Reestimation model parameters for vowel "ϵ (bet)"

$\pi_i =$

$1.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$

$a_{ij} =$

$9.54210e-01$	$2.61651e-02$	$9.81194-e02$	$5.72363-e-03$	$4.08830e-03$
$0.00000e-00$	$5.71423e-01$	$2.14882e-01$	$1.25001e-01$	$8.92867e-02$
$0.00000e-00$	$0.00000e-00$	$4.99993e-01$	$2.91666e-01$	$2.08333e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$5.83333e-01$	$4.16667e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$1.00000e-00$

b_{ik} =$9.99995e-01$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.9: Reestimation model parameters for vowel "$\wedge$ (but)"

$\pi_i =$

$1.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$

$a_{ij} =$

$9.54210e-01$	$2.61651e-02$	$9.81194-e02$	$5.72363-e-03$	$4.08830e-03$
$0.00000e-00$	$5.71423e-01$	$2.14882e-01$	$1.25001e-01$	$8.92867e-02$
$0.00000e-00$	$0.00000e-00$	$4.99993e-01$	$2.91666e-01$	$2.08333e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$5.83333e-01$	$4.16667e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$1.00000e-00$

$b_{ik} =$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.10: Reestimation model parameters for vowel "a (hot)"

$\pi_i =$

$1.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$

$a_{ij} =$

$9.54210e-01$	$2.61651e-02$	$9.81194-e02$	$5.72363-e-03$	$4.08830e-03$
$0.00000e-00$	$5.71423e-01$	$2.14882e-01$	$1.25001e-01$	$8.92867e-02$
$0.00000e-00$	$0.00000e-00$	$4.99993e-01$	$2.91666e-01$	$2.08333e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$5.83333e-01$	$4.16667e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$1.00000e-00$

$b_{ik} =$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.11: Reestimation model parameters for vowel "I (bit)"

$\pi_i =$

$1.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$

$a_{ij} =$

$9.54210e-01$	$2.61651e-02$	$9.81194-e02$	$5.72363-e-03$	$4.08830e-03$
$0.00000e-00$	$5.71423e-01$	$2.14882e-01$	$1.25001e-01$	$8.92867e-02$
$0.00000e-00$	$0.00000e-00$	$4.99993e-01$	$2.91666e-01$	$2.08333e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$5.83333e-01$	$4.16667e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$1.00000e-00$

$b_{ik} =$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$

Table 4.12: Reestimation model parameters for vowel "O (obey)"

$\pi_i =$

$1.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$

$a_{ij} =$

$9.54210e-01$	$2.61651e-02$	$9.81194-e02$	$5.72363-e-03$	$4.08830e-03$
$0.00000e-00$	$5.71423e-01$	$2.14882e-01$	$1.25001e-01$	$8.92867e-02$
$0.00000e-00$	$0.00000e-00$	$4.99993e-01$	$2.91666e-01$	$2.08333e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$5.83333e-01$	$4.16667e-01$
$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$0.00000e-00$	$1.00000e-00$

$b_{ik} =$

$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$	$1.00000e-05$
$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$	$9.99995e-01$

Table 4.13: Reestimation model parameters for vowel "U (food)"

Conclusion:

Isolated word recognition system based on HMM is implemented for the recognition of six vowels (namely ϵ (bet), $\wedge$(but), a (hot), I (bit), O (obey), U (food)). While simulating the system, it is observed that the most important part is the estimation of model parameters. Performance of system depends upon the value of model parameters. The computed values of models parameters give satisfactory result. Approximately 90% of recognition accuracy has been obtained . Performance of system can further be enhanced by using large training data because the HMM model estimation algorithms give better estimation of the HMM parameters for large training data. But recording and digitization of speech is a lengthy process, hence we have restricted our work for small training data. The determined model parameters for left-right type HMM gives an accuracy of about 92.45%.

Greater the number of states, more fine details of speech can be modeled. The five states are sufficient for our purpose. The states more than five do not lead to significant improvement in performance[6]. After analyzing the state symbol probability matrix, it is found that each cluster represents one specific vowel.

Appendix A

Dynamic Time Warping Algorithm

In case of speech recognition system based on pattern comparison technique, similarity measurement between the reference pattern and test pattern is an important issue. In such system time registration of test pattern and reference pattern is important because time scales of test pattern and reference pattern are generally not perfectly aligned. Time duration for test pattern is different from duration of reference pattern. Hence the importance of DTW lies in compression or expansion of test pattern so that the similarity can be measured effectively. The mathematical framework for general DTW algorithm is explained here.

Specification Of The DTW Algorithm

It was assumed that end point of test pattern and reference pattern is known prior to DTW algorithm applied. Let the test pattern is denoted as

$$T = \{T(1), T(2) \ldots . T(M)\} \tag{A.1}$$

where $T(m)$ is spectral vector of input speech at a time m and M is the total number of frames of speech. Similarly a set of reference pattern $\{R^1, R^2, \ldots, R^V\}$, where each reference pattern R^j is denoted as

$$R^j = \{R(1), R(2), \ldots . R(N)\} \tag{A.2}$$

where $R(n)$ is spectral vector at time n and N is total number of frames of speech pattern.

The purpose of DTW algorithm is to find the path

$$m = w(n) \tag{A.3}$$

in (n, m) plane which is an optimal path i.e which minimizes the total distance function D. The function D is given by following formula:

$$D = \sum_{n=1}^{N} \widehat{d}(R(n), T(w(n))) \tag{A.4}$$

where $\widehat{d}(R(n), T(w(n)))$ is the local distance between frame of reference pattern and frame $[m = w(n)]$ of the test pattern.

Let express both time axes (n, m) in term of common time axis k as

$$n = i(k) \quad k = 1, 2, \ldots, K \tag{A.5}$$

$$m = j(k) \quad k = 1, 2, \ldots, K \tag{A.6}$$

where K is length of the common time axis. In order to find the best path in the (n, m) plane following factors of DTW must be considered[11].

1. End Point Constraints:

In isolated word recognition system, the test pattern and reference pattern have well defined end points that mark the beginning and the ending of frames of pattern. The end point constraint is of the form:

$$i(1) = 1, \quad j(1) = 1, \quad \textit{beginning point} \tag{A.7}$$

$$i(k) = N \quad j(k) = M, \quad \textit{ending point} \tag{A.8}$$

2.Local Continuity Constraint

To ensure proper time alignment and to avoid excessive loss of information, a set of local continuity constraint is applied. A first constraint of type is the monotonicity constraint, namely

$$i(k+1) \geq i(k) \tag{A.9}$$

$$j(k+1) \geq j(k) \tag{A.10}$$

The path P_r is defined as sequence of moves, each specified by a pair of co-ordinate increments

$$P_r \rightarrow (\alpha_1^{(r)}, \beta_1^{(r)})\,(\alpha_2^{(r)}, \beta_2^{(r)}) \ldots . (\alpha_{L(r)}^{(r)}, \beta_{L(r)}^{(r)}) \tag{A.11}$$

where r signifies the rth path and $L(r)$ denotes length of rth path. The path is traced in backward direction through $L(r)$ points as

$$k\,th\,point: \; i(k) = n, \qquad j(k) = m \tag{A.12}$$

$$(k-s)th\,point:\ i(k-s) = i(k) - \sum_{l=1}^{s} \alpha_l^{(r)} \tag{A.13}$$

$$j(k-s) = j(k) - \sum_{l=1}^{s} \beta_l^{(r)} \tag{A.14}$$

for $s = 1, 2, \ldots L(r)$.

3.Global Path constraints:

These constraints resist the path to lie in some region of (n,m) plane. The boundary of such region in which optimal path lies, is given by following inequality

$$1 + \frac{(i(k)-1)}{E_{max}} \leq j(k) \leq 1 + E_{max}(i(k)-1) \tag{A.15}$$

$$M + E_{max}(i(k)-N) \leq j(k) \leq M + \frac{(i(k)-N)}{E_{max}} \tag{A.16}$$

where

$$E_{max} = \max_{(r)} \left[\frac{\sum_{l=1}^{L(r)} \beta_l^{(r)}}{\sum_{l=1}^{L(r)} \alpha_l^{(r)}} \right] \tag{A.17}$$

$$E_{max} = \min_{(r)} \left[\frac{\sum_{l=1}^{L(r)} \beta_l^{(r)}}{\sum_{l=1}^{L(r)} \alpha_l^{(r)}} \right] \tag{A.18}$$

4.Axis Orientation

Previously we have defined a common time axis k and arbitrary n to $i(k)$ and m to $j(k)$. We can also assign

$$n = j(k), \qquad k = 1, 2, \ldots . K \tag{A.19}$$

$$m = i(k), \qquad k = 1, 2, \ldots . K. \tag{A.20}$$

There is no difference between above assignment and previous assignment when both the local constraints and distance metric are symmetric. However when there is asymmetry in either local constraints or in distance metric, then the difference in variable assignment can be significant.

5.Distance Measure

The last factor in the DTW is the distance measure between reference pattern and test pattern. The distance function is defined as

$$D(i(k),j(k)) = \frac{\sum_{k=1}^{K} d(i(k),j(k))\widetilde{W}(k)}{N(\widetilde{W})} \tag{A.21}$$

where $D(i(k),j(k))$ is a function that gives the total distance along the path of length K. $d(i(k),j(k))$ is the local distance between frame $i(k)$ of the reference, and $j(k)$ of the test. $\widetilde{W}(k)$ is a weighing function of the kth arc of the path, and $N(\widetilde{W})$ is a normalization factor which is a function of the weighing function $\widetilde{W}$.

The distance computed along the optimal path is minimum. Let this minimum distance is denoted as $\widehat{D}$, then

$$\widehat{D} = \min_{(K,i(k),j(k))} (D(i(k),\, j(k))). \tag{A.22}$$

For the implementation of $\widehat{D}$ the local distance function d, the weighing function $\widetilde{W}$, and the normalized factor $N(\widetilde{W})$ must be specified.

The complete isolated word recognition system based on DTW algorithm is shown in figA.1.

The front end processor provides a set of $(p+1)$ autocorrelation coefficient for each frame. The output of the DTW algorithm for the vth reference word

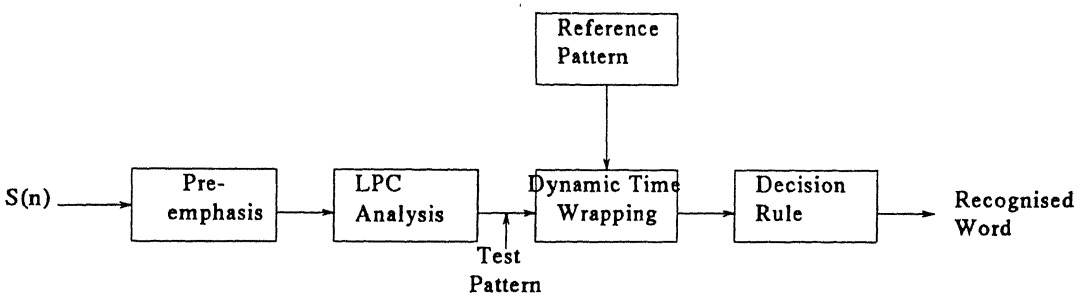


Figure A.1: Block Diagram of LPC based word recognizer using a standard DTW algorithm.

is the distance $\widehat{D}^{(v)}$, $v = 1, 2, .R$, and the decision rule processes the set of $\widehat{D}^{(v)}$. It selects the reference pattern which gives smallest value of $\widehat{D}$.

Appendix B

Linear Predictive Coding Of Speech

Linear predictive analysis[13] is one of the most powerful speech signal analysis technique. It provides extremely accurate estimation of speech parameters (e.g. pitch, formats, spectra, vocal tract area function). The basic idea behind LPC modeling is that speech sample at time n , $s(n)$ can be approximated as a linear combination of past p samples i.e., $s(n)$ can be computed as

$$s(n) = a_1 s(n-1) + a_2 s(n-2) + \ldots\ldots + a_p s(n-p) \qquad \text{(B.1)}$$

where the coefficients $a_1, a_2, \ldots . a_p$ are assumed to be constant over speech analysis frame. Based on the model shown in figB.1. the relation between $s(n)$ and $u(n)$ (normalized exited source) is given as

$$s(n) = \sum_{k=1}^{p} a_k s(n-k) \; + Gu(n) \qquad \text{(B.2)}$$

The estimated $\tilde{s}(n)$ is defined as:

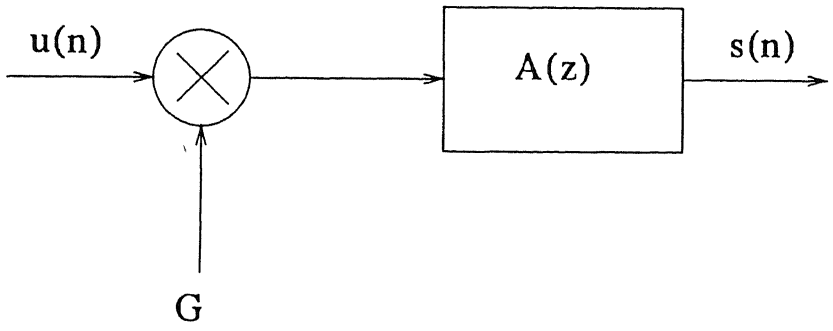


Figure B.1: Linear Predictive Model for Speech.

$$\tilde{s}(n) = \sum_{k=1}^{p} a_k s(n-k). \tag{B.3}$$

The prediction error $e(n)$ is calculated as

$$e(n) = s(n) - \tilde{s}(n) = s(n) - \sum_{k=1}^{p} a_k s(n-k). \tag{B.4}$$

Now we have to determine a set of predictor coefficients $\{a_k\}$ directly from the speech in such a manner so as to obtain good estimate of spectral properties of speech signal. The basic approach is to find a set of predictor coefficients that will minimize the mean square prediction error over a short segment of the speech waveform.

Let the short time average prediction error is defined as

$$E_n = \sum_m e_n^2(m) \tag{B.5}$$

$$\begin{aligned} &= \textstyle\sum_m [s_n(m) - \tilde{s}_n(m)]^2 \\ &= \textstyle\sum_m [s_n(m) - \sum_{k=1}^{p} a_k s_n(m-k)]^2 \end{aligned} \tag{B.6}$$

where $s_n(m)$ is a segment of speech that has been selected in the vicinity of the sample n i.e.

$$s_n(m) = s(m+n) \tag{B.7}$$

We can find the value of a_k that minimizes E_n in equation B.4 by setting

$$\frac{\partial E_n}{\partial a_i} = 0 \quad i = 1, 2, \ldots . p \tag{B.8}$$

thereby obtaining

$$\phi_n(i,k) = \sum_{k=1}^{p} a_k \phi_n(i,k) \quad i = 1, 2, \ldots . p \tag{B.9}$$

where $\phi(i,k)$ is defined as

$$\phi_n(i,k) = \sum_{m} s_n(m-i) s_n(m-k) \tag{B.10}$$

To solve eqB.9 for the optimum prediction coefficients we have to compute $\phi_n(i,k)$ for $1 \leq i \leq p$ and $0 \leq k \leq p$ and then solve the resulting set of p simultaneous equations. There are two standard methods for solving.

1.The autocorrelation method

In this approach $s_n(m)$ is assumed to be zero outside the interval $0 \leq m \leq N-1$. Mathematically

$$s_n(m) = s(m+n) * w(m) \tag{B.11}$$

where $w(m)$ is finite length window (e.g Hamming Window) that is identically zero outside the interval $0 \leq m \leq N-1$. $s_n(m)$ is non zero only for $0 \leq m \leq N-1$ and the corresponding error, $e_n(m)$ for the p^{th} order predictor will be non

zero over the interval $0 \leq m \leq N-1+p$. Thus for this case E_n is expressed as

$$E_n = \sum_{m=0}^{N+p-1} e_n^2(m) \tag{B.12}$$

and eqB.7 becomes

$$\phi_n(i,k) = \sum_{m=0}^{N+p-1} s_n(m-i)s_n(m-k) \; {}^{1\leq i\leq p}_{0\leq k\leq p} \tag{B.13}$$

it can be reduced to

$$\phi_n(i,k) = \sum_{m=0}^{N-1-(i-k)} s_n(m)s_n(m+i-k) \; {}^{0\leq i\leq p}_{0\leq k\leq p} \tag{B.14}$$

In this case $\phi_n(i,k)$ is identical to the short time autocorrelation function evaluated for (i,k). That is

$$\phi_n(i,k) = R_n(i-k) \tag{B.15}$$

where

$$R_n = \sum_{m=0}^{N-1-k} s_n(m)s_n(m+k) \tag{B.16}$$

Therefor eq B.9 represented as

$$\sum_{k=1}^{p} a_k R_n(|i-k|) = R_n(i) \quad 1 \leq i \leq p \tag{B.17}$$

It can be represented in matrix form as

$$\begin{bmatrix} R_n(0) & R_n(1) & R_n(2) & .. & .. & R_n(p-1) \\ R_n(1) & R_n(0) & R_n(1) & .. & .. & R_n(p-2) \\ R_n(2) & R_n(1) & R_n(0) & .. & .. & R_n(p-3) \\ .... & .... & .... & .. & .. & .... \\ .... & .... & .... & .. & .. & .... \\ R_n(p-1) & R_n(p-2) & R_n(p-3) & .. & .. & R_n(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ .. \\ .. \\ a_p \end{bmatrix} = \begin{bmatrix} R_n(1) \\ R_n(2) \\ R_n(3) \\ ... \\ ... \\ R_n(p) \end{bmatrix} \tag{B.18}$$

The $p \times p$ matrix of autocorrelation value is a Toeplize matrix, i.e it is symmetric and all elements along a given diagonal are equal.

2.The covariance method

The second basic approach to define the speech segment $s_n(m)$ and limits on the sum, is to fix the interval over which the mean square error is computed. Then if we define

$$E_n = \sum_{m=0}^{N-1} e_n^2(m) \tag{B.19}$$

then $\phi_n(i,k)$ becomes

$$\phi_n(i,k) = \sum_{m=0}^{N-1} s_n(m-i)s_n(m-k) \quad \begin{matrix} 0 \le i \le p \\ 0 \le k \le p \end{matrix} \tag{B.20}$$

or by change of variable

$$\phi_n(i,k) = \sum_{m=0}^{N-1-(i-k)} s_n(m)s_n(m+i-k) \quad \begin{matrix} 0 \le i \le p \\ 0 \le k \le p \end{matrix} \tag{B.21}$$

Now eqB.7 becomes

$$\phi_n(i,0) = \sum_{k=1}^{p} a_k \phi_n(i,k) \quad i = 1, 2, \ldots . p \tag{B.22}$$

in matrix form it can be represented as

$$\begin{bmatrix} \phi_n(1,1) & \phi_n(1,2) & \phi_n(1,3) & .. & .. & \phi_n(1,p) \\ \phi_n(2,1) & \phi_n(2,2) & \phi_n(2,3) & .. & .. & \phi_n(2,p) \\ \phi_n(3,1) & \phi_n(3,2) & \phi_n(3,3) & .. & .. & \phi_n(3,p) \\ .... & .... & .... & .. & .. & .... \\ .... & .... & .... & .. & .. & .... \\ \phi_n(p,1) & \phi_n(p,2) & \phi_n(p,3) & .. & .. & \phi_n(p,p) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ .. \\ .. \\ a_p \end{bmatrix} = \begin{bmatrix} \phi_n(1,0) \\ \phi_n(2,0) \\ \phi_n(3,0) \\ .... \\ .... \\ \phi_n(p,0) \end{bmatrix} \tag{B.23}$$

The resulting matrix can be solved by Cholesky Decomposition.

Here we are interested in the solution of autocorrelation matrix.

3.Durbin's recursive solution for the autocorrelation equation

For autocorrelation method the matrix equationB.18 can be solved by several efficient recursion procedures. Durbin's recursive procedure is given as

$E^{(0)} = R(0)$

$k_i = \left[R(i) - \sum_{j=1}^{i-1} a_j^{(i-1)} R(i-j)\right] / E^{(i-1)} \quad 1 \le i \le p$

$a_i^{(i)} = k_i$

$a_j^{(i)} = a_j^{(i-1)} - k_i * a_{i-j}^{(i-1)} \qquad i \le j \le i-1$

$E^{(i)} = (1 - k_i^2) E^{(i-1)}$

Above equations are solved recursively for $i = 1, 2, \ldots p$ and the final solution is given as

$$a_j = a_j^{(p)} \quad 1 \le j \le p. \tag{B.24}$$

Durbin's recursive procedure is the most efficient procedure known for solving this system of equations.

Bibliography

[1] L.R.Rabiner and R.W.Schofer "Digital Processing of Speech Signal" Prentice Hall Inc. 1978

[2] L.R.Rabiner "A Tutorial on Hidden Markov Modeling and Selected Application in Speech Recognition" Proceeding of the IEEE, vol 77, no.2, pp 257-285, February 1989.

[3] L.R.Rabiner and B.H.Juang, "An Introduction to Hidden Markov Modeling" IEEE ASSP Magazine, January 1986.

[4] L.Rabiner and B.H.Juang, "Fundamentals of Speech Recognition" Prentice Hall, 1994.

[5] A.Weibel, T.Hanazawa, G.Hinton, K.Shikano, and K.J.Lang, "Phoneme Recognition Using Time Delay Neural Networks," IEEE Trans, ASSP 37pp 328-339, 1989.

[6] L.R.Rabiner, S.E.Levinsion, M.M.Sondi "On The Application of Vector Quantization and Hidden Markov Models to Speaker Independent, Isolated Word Recognition," The BELL, System Technical Journal, April 1983, pp 1075-1105.

[7] Hui-Ling Lou "Implementing the Viterbi Algorithm" IEEE Signal Processing Magazine, September 1995 pp 42-52.

[8] G.David Forney, Jr "The Viterbi Algorithm," Proc. of the IEEE, vol 61no 3, March 1973 pp 268-278.

[9] S.Furi and M.M.Sondi, "Advance in Speech Signal Processing" Marcel Dekker Inc New York 1992

[10] Richard. A. Johnson and Wichern, "Applied Multivarient statistical Analysis". Prentice Hall 1988 pp 543-573.

[11] C.Myers, L.R.Rabiner and A.E.Rosenberg, "Performance tradeoff in Dynamic Time Warping Algorithm for isolated word recognition" IEEE transaction Acoustic, speech & signal Processing, vol ASSP 28, No 6. Dec-1980.

[12] C.S.Mayer and L.R.Rabiner, "A comparative Study of several Dynamic Time Warping Algorithm for Connected word recognition," The BELL System Technical Journal vol 60, No 7, September 1981.

[13] Rabiner and R.W.Schofer, "Digital Processing of Speech Signal" Prentice Hall Inc. 1978 pp 396-453.

[14] L.R.Rabiner and M.R.Sambur, "An Algorithm for Determining the Endpoints of Isolated Utterances," The BELL System Technical Journal vol 54, No 2, February 1975.

[15] L.F.Lamb, L.R.Rabiner, A.E.Rosemberg and t.g.Wilpon, "An improved of endpoint Detector for isolated word recognition" IEEE Trans. ASSP vol 29 No 4 pp 777-785 Aug. 1981.

[16] Jae.S.Lim, "Two dimensional Signal and image Processing," Prentice Hall, pp 598-603.

[17] F.Itakura, "Minimum prediction residual principle applied to speech recognition" IEEE trans. Acoustic speech signal Proc., ASSP vol 23 pp 57-72 February 1975.

A 130786

Date Slip

This book is to be returned on the date last stamped.

.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....